R 言語による決定木構築

1 データの読み込み

- 今回は，配布データ zoo.csv を利用する．このデータは，http://archive.ics.uci.edu/ml/datasets/zoo から入手したデータを，一部加工したものである．動物の分類に関するデータであり，データ数は 100，属性数は 17 である．属性”type”がクラスを表す．また type の値は，amphibian(両生類), bird(鳥類), fish(魚類), insect(昆虫), invertebrate(無脊椎動物), mammal(哺乳類), reptile(爬虫類) である．詳細は当該サイトを参照のこと．

```
> zoo <- read.csv("zoo.csv", strip.white=TRUE,row.names=1)
>
> dim(zoo)
[1] 100  17
>
> colnames(zoo) #属性名の表示
 [1] "hair"     "feathers" "eggs"     "milk"     "airborne" "aquatic"  "predator" "toothed"  "backbone"
[10] "breathes" "venomous" "fins"     "legs"     "tail"     "domestic" "catsize"  "type"
>
> summary(zoo$type) #クラスごとの事例数の表示
   amphibian         bird         fish       insect invertebrate       mammal      reptile
           3           20           13            8           10           41            5
>
```

2 決定木の構築

● ライブラリの読み込み

- 今回はライブラリ，rpart を用いる．ライブラリのロードは，library コマンドで行う．

```
> library(rpart)
```

- rpart は標準でインストールされているが，もしインストールされてない場合は，以下のコマンドでインストールすること．

```
> install.packages("rpart", dependencies=TRUE)
```

● 決定木の構築

- 決定木構築コマンドは，”rpart”である．
- このコマンドは，第一引数として，クラス変数（目的変数）と利用する属性（説明変数）を，

クラス変数名~.

もしくは，

クラス変数名~属性 1 +属性 2 +···+属性 3

の形式で指定する (具体例は後述)．なお，”.”(ピリオド) は，すべての説明変数を意味する．

- ”rpart”コマンドの第二引数では，利用するデータを指定する．
- また，返り値として得られる決定木を，適当な名前の変数に保存することをお勧めします．

● 決定木の表示を読み方

- 構築した決定木は，木を格納した”変数名”を入力することで，表示することが出来る．[1]
- 結果表示の 1 行目は，データ数である．
- 各ノードは，二行目にある　node)，split，n，loss，yval，(yprob)　の形式で表現される．
  - 　node) ノード番号を表す．ex:　2)
  - split 分割のための条件を表す．ex:　milk=false
  - n ノードにマッチする訓練事例数．ex:59
  - loss 誤分類事例数．ex:39
  - yval 予測クラス．ex:bird
  - yprob クラス分布．ex:(0.03 0.2 0.13 0.08 0.1 0.41 0.05)
- summary コマンドを利用すると，より詳細な情報が表示される．
- 分割表の作成・表示には，table コマンドを利用する．
- 以下に，一連の操作の例を示す．

```
> library(rpart)  #ライブラリの読み込み
>
> zoo.dt <- rpart(type ~ ., zoo)  #全属性を利用して，決定木を構築．結果を，zoo.dt へ代入
>
> zoo.dt_milk_eggs <- rpart(type ~ milk + eggs, zoo) #属性 milk，eggs を利用して分類器を構築
>
> zoo.dt  #全属性を利用した場合の決定木を表示
n= 100

node), split, n, loss, yval, (yprob)
* denotes terminal node

1) root 100 59 mammal (0.03 0.2 0.13 0.08 0.1 0.41 0.05)
  2) milk=false 59 39 bird (0.051 0.34 0.22 0.14 0.17 0 0.085)
    4) feathers=true 20  0 bird (0 1 0 0 0 0 0) *
    5) feathers=false 39 26 fish (0.077 0 0.33 0.21 0.26 0 0.13)
      10) fins=true 13  0 fish (0 0 1 0 0 0 0) *
      11) fins=false 26 16 invertebrate (0.12 0 0 0.31 0.38 0 0.19)
        22) backbone=false 18  8 invertebrate (0 0 0 0.44 0.56 0 0) *
        23) backbone=true 8  3 reptile (0.38 0 0 0 0 0 0.62) *
  3) milk=true 41  0 mammal (0 0 0 0 0 1 0) *
>
> zoo.dt_milk_eggs  #属性 milk と eggs で構築された分類器を表示
n= 100

node), split, n, loss, yval, (yprob)
      * denotes terminal node

1) root 100 59 mammal (0.03 0.2 0.13 0.08 0.1 0.41 0.05)
  2) milk=false 59 39 bird (0.051 0.34 0.22 0.14 0.17 0 0.085) *
  3) milk=true 41  0 mammal (0 0 0 0 0 1 0) *
>
>
> table(zoo$type, predict(zoo.dt, zoo, type="class")) #分割表の表示
# 各行 (table の第一引数) が実際のクラス，各列（table の第二引数）が予測クラス
              amphibian bird fish insect invertebrate mammal reptile
  amphibian           0    0    0      0           0      0       3
  bird                0   20    0      0           0      0       0
  fish                0    0   13      0           0      0       0
  insect              0    0    0      0           8      0       0
  invertebrate        0    0    0      0          10      0       0
  mammal              0    0    0      0           0     41       0
  reptile             0    0    0      0           0      0       5
>
```

[1] partykit ライブラリなどをインストールすると，より視覚的に結果を表示することが出来る．興味のある学生さんは是非トライしてみよう．

3 オプションの指定

● 分割基準の設定

- rpart では，分割基準としてジニ係数と情報利得を利用することが出来る．具体的には，以下の書式でパラメタ split を指定する．

```
parms = list(split="information_or_gini")
```

```
> rpart(type ~ . , zoo, parms=list(split="information")) #情報利得を用いた決定木
n= 100

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 100 59 mammal (0.03 0.2 0.13 0.08 0.1 0.41 0.05)
   2) milk=false 59 39 bird (0.051 0.34 0.22 0.14 0.17 0 0.085)
     4) feathers=true 20  0 bird (0 1 0 0 0 0 0) *
     5) feathers=false 39 26 fish (0.077 0 0.33 0.21 0.26 0 0.13)
      10) backbone=true 21  8 fish (0.14 0 0.62 0 0 0 0.24)
        20) fins=true 13  0 fish (0 0 1 0 0 0 0) *
        21) fins=false 8  3 reptile (0.38 0 0 0 0 0 0.62) *
      11) backbone=false 18  8 invertebrate (0 0 0 0.44 0.56 0 0) *
   3) milk=true 41  0 mammal (0 0 0 0 0 1 0) *
>
> rpart(type ~ . , zoo, parms=list(split="gini"))  #gini 係数を用いた決定木（デフォルト）
n= 100

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 100 59 mammal (0.03 0.2 0.13 0.08 0.1 0.41 0.05)
   2) milk=false 59 39 bird (0.051 0.34 0.22 0.14 0.17 0 0.085)
     4) feathers=true 20  0 bird (0 1 0 0 0 0 0) *
     5) feathers=false 39 26 fish (0.077 0 0.33 0.21 0.26 0 0.13)
      10) fins=true 13  0 fish (0 0 1 0 0 0 0) *
      11) fins=false 26 16 invertebrate (0.12 0 0 0.31 0.38 0 0.19)
        22) backbone=false 18  8 invertebrate (0 0 0 0.44 0.56 0 0) *
        23) backbone=true 8  3 reptile (0.38 0 0 0 0 0 0.62) *
   3) milk=true 41  0 mammal (0 0 0 0 0 1 0) *
>
```

● 枝刈基準の設定

- control に値を指定することで，枝刈り等の基準を設定することが出来る．

```
control = rpart.control(パラメタ名=値のリスト)
```

- ノードに含まれる最低数を設定するには，minsplit を指定する (デフォルト値 20)．
- 木の高さを限定するには maxdepth を指定する (デフォルト値 30)．
- コスト複雑度枝刈りのパラメタ (α) に相当するパラメタは cp である．値が小さいほど枝刈りが行われず，0 の場合には全く枝刈を行わない (デフォルト値 0.01)．
- 以下に実行例を示す

```
#ノードに含まれる最低数を 10 に設定
> rpart(type ~ . , zoo, parms=list(split="gini"), control=rpart.control(minsplit=10))
n= 100

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 100 59 mammal (0.03 0.2 0.13 0.08 0.1 0.41 0.05)
   2) milk=false 59 39 bird (0.051 0.34 0.22 0.14 0.17 0 0.085)
     4) feathers=true 20  0 bird (0 1 0 0 0 0 0) *
     5) feathers=false 39 26 fish (0.077 0 0.33 0.21 0.26 0 0.13)
      10) fins=true 13  0 fish (0 0 1 0 0 0 0) *
      11) fins=false 26 16 invertebrate (0.12 0 0 0.31 0.38 0 0.19)
        22) backbone=false 18  8 invertebrate (0 0 0 0.44 0.56 0 0)
          44) airborne=true 6  0 insect (0 0 0 1 0 0 0) *
          45) airborne=false 12  2 invertebrate (0 0 0 0.17 0.83 0 0) *
        23) backbone=true 8  3 reptile (0.38 0 0 0 0 0 0.62) *
   3) milk=true 41  0 mammal (0 0 0 0 0 1 0) *

# 木の高さを，2 に設定
> rpart(type ~ . , zoo, parms=list(split="gini"), control=rpart.control(maxdepth=2))
n= 100

node), split, n, loss, yval, (yprob)
      * denotes terminal node

1) root 100 59 mammal (0.03 0.2 0.13 0.08 0.1 0.41 0.05)
  2) milk=false 59 39 bird (0.051 0.34 0.22 0.14 0.17 0 0.085)
    4) feathers=true 20  0 bird (0 1 0 0 0 0 0) *
    5) feathers=false 39 26 fish (0.077 0 0.33 0.21 0.26 0 0.13) *
  3) milk=true 41  0 mammal (0 0 0 0 0 1 0) *
>

#複数のパラメタを組み合わせるには，それぞれをカンマで区切って並べる.
> rpart(type ~ . , zoo, parms=list(split="gini"), control=rpart.control(cp=0,minsplit=1,maxdepth=30))
n= 100

node), split, n, loss, yval, (yprob)
      * denotes terminal node

  1) root 100 59 mammal (0.03 0.2 0.13 0.08 0.1 0.41 0.05)
    2) milk=false 59 39 bird (0.051 0.34 0.22 0.14 0.17 0 0.085)
      4) feathers=true 20  0 bird (0 1 0 0 0 0 0) *
      5) feathers=false 39 26 fish (0.077 0 0.33 0.21 0.26 0 0.13)
       10) fins=true 13  0 fish (0 0 1 0 0 0 0) *
       11) fins=false 26 16 invertebrate (0.12 0 0 0.31 0.38 0 0.19)
         22) backbone=false 18  8 invertebrate (0 0 0 0.44 0.56 0 0)
           44) airborne=true 6  0 insect (0 0 0 1 0 0 0) *
           45) airborne=false 12  2 invertebrate (0 0 0 0.17 0.83 0 0)
             90) predator=false 4  2 insect (0 0 0 0.5 0.5 0 0)
              180) legs>=3 2  0 insect (0 0 0 1 0 0 0) *
              181) legs< 3 2  0 invertebrate (0 0 0 0 1 0 0) *
             91) predator=true 8  0 invertebrate (0 0 0 0 1 0 0) *
         23) backbone=true 8  3 reptile (0.38 0 0 0 0 0 0.62)
           46) aquatic=true 4  1 amphibian (0.75 0 0 0 0 0 0.25)
             92) eggs=true 3  0 amphibian (1 0 0 0 0 0 0) *
             93) eggs=false 1  0 reptile (0 0 0 0 0 0 1) *
           47) aquatic=false 4  0 reptile (0 0 0 0 0 0 1) *
    3) milk=true 41  0 mammal (0 0 0 0 0 1 0) *
>
```

4 決定木の評価

● テストデータを用いた評価

- 決定木の主たる目的の一つは，新たなデータのクラスを予測することである．ここでは，以下の手順に沿って，決定木の構築・評価を行う．

(1) データを，訓練例集合とテスト例集合に分割する
(2) 訓練例集合を用いて，決定木を学習する
(3) 得られた決定木を用いて，テスト例集合のクラスを予測する
(4) テスト例集合の実クラスと予測クラスを用いて，分割表を作成する
(5) χ^2 検定を行う

(1) データを訓練例集合とテスト例集合に分割する： CSV ファイルを作る段階で，訓練例とテスト例に分けるのが簡単である．その一方で，R 上で，データを分割したいことも考えられる．今回は，後者（R 上でのデータの分割）の方法を紹介する．

```
> dim(zoo)      #データ zoo を分割する.
[1] 100  18
> zoo.train <- zoo[1:80, ]  #zoo の 1〜80 行目までを取り出し，zoo.train へ格納
> zoo.test <- zoo[81:100, ] #zoo の 81〜100 行目までを取り出し，zoo.test へ格納
>
> dim(zoo.train)    #データ数を確認
[1] 80 17
> dim(zoo.test)     #データ数を確認
[1] 20 17
```

(2) 訓練例から決定木を構築する： 既に説明した通り，”rpart”コマンドを用いて，決定木を構築する．その際，データとして訓練例集合を指定する．下記の例では，訓練例集合 zoo.train を対象に，クラス属性”type”を予測する決定木を構築している．

```
> zoo.train.dt <- rpart( type ~ ., zoo.train )
> zoo.train.dt
n= 80

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 80 44 mammal (0.025 0.2 0.12 0.075 0.087 0.45 0.037)
   2) milk=false 44 28 bird (0.045 0.36 0.23 0.14 0.16 0 0.068)
     4) feathers=true 16  0 bird (0 1 0 0 0 0 0) *
     5) feathers=false 28 18 fish (0.071 0 0.36 0.21 0.25 0 0.11)
      10) fins=true 10  0 fish (0 0 1 0 0 0 0) *
      11) fins=false 18 11 invertebrate (0.11 0 0 0.33 0.39 0 0.17) *
   3) milk=true 36  0 mammal (0 0 0 0 0 1 0) *
>
```

(3) 得られた決定木を用いて，テスト例集合のクラスを予測する： クラスの予測には，”predict”コマンドを使う．このコマンドは，第一引数に学習した決定木を，第二引数に予測したテスト例の集合を指定する．加えて，type の値として class を指定する [2]．

```
predict(決定木, テスト例集合, type="class" )
```

結果は，適当な変数に格納しておくと便利である．以下の例では，学習済みの決定木”zoo.train”を使って，テスト例集合”zoo.test”の各データのクラスを予測し，その結果を変数”zoo.test.predict”に格納しています．

```
> zoo.test.predict <- predict(zoo.train.dt, zoo.test, type="class")
```

(4) 分割表の作成： テスト例集合の実クラスと予測クラスを用いて，分割表を作成する．この操作には，”table”コマンドを利用する．このコマンドは,「実クラスの集合」と「予測クラスの集合」を引数にとる．また，結果は適当な変数に保存する．

```
> result <- table(zoo.test$type, zoo.test.predict )

# 各行 (table の第一引数) が実際のクラス，各列（table の第二引数）が予測クラス
> result
              zoo.test.predict
               amphibian bird fish insect invertebrate mammal reptile
  amphibian            0    0    0      0            1      0       0
  bird                 0    4    0      0            0      0       0
  fish                 0    0    3      0            0      0       0
  insect               0    0    0      0            2      0       0
  invertebrate         0    0    0      0            3      0       0
  mammal               0    0    0      0            0      5       0
  reptile              0    0    0      0            2      0       0
>
```

上記の例で，”table”コマンドの第一引数”zoo.test$type”は，テスト例集合「zoo.test の"type"列を抽出

[2] ここでの type は，目的変数の type とは無関係

せよ」という意味である．利用するデータに合わせて，適切な列を選択すること．

また，上記の分割表では，

- 各行が実クラス（table の第一引数）
- 各列が予測クラス（table の第二引数）

に，それぞれ対応します．例えば上記の例では，テスト例集合中に，クラスが”reptile”と予測された例はないことが分かる．

(5)χ^2 検定の実施：”chisq.test”コマンドを用いて χ^2 検定を行うことができる．このコマンドの引数には，分割表を指定する．

```
> chisq.test(result)

        Pearson's Chi-squared test

data:  result
X-squared = NaN, df = 36, p-value = NA
```

検定結果を解釈するには，”p-value”の値に着目する．p-value(p 値) とは，帰無仮説（実クラスと予測クラスが独立）のもとで，この結果 (χ^2 統計推定量) が偶然得られる確率を表す．この値が，例えば 5%よりも小さければ，有意水準 5%で帰無仮説を棄却することができる．

ところで，今回の例のように，データによっては正しく χ^2 値が計算できない場合がある．今回は，”reptile”と予測されたデータがなかったため，χ^2 値の計算過程で “0 除算” が生じたことが原因である．本来的な原因は,「訓練例集合とテスト例集合の作り方が悪い」ということであり，何回かデータを作り直すのが，正しい対処方法となる．

一方，原因は “0 除算” なので，かなりアドホックではあるが，分割表全体に非常に小さな値を追加するというのも手かもしれない（繰り返しますが，かなりアドホックです）

```
> chisq.test( result+0.00001 )

        Pearson's Chi-squared test

data:  result + 1e-05
X-squared = 59.999, df = 36, p-value = 0.007272

 警告メッセージ:
 chisq.test(result + 1e-05) で:   カイ自乗近似は不正確かもしれません
>
```